GOOGLE PLAYSTORE APPS ANALYSIS

A PROJECT REPORT

Submitted by

BL.EN.U4AIE19041 M. TANUJ

BL.EN.U4AIE19047 P. VENKATA AKHIL

BL.EN.U4AIE19068 AISHWARYA.V

in partial fulfillment for the award of the degree

0f

BACHELOR OF TECHNOLOGY

IN

ARTIFICIAL INTELLIGENCE ENGINEERING

AMRITA SCHOOL OF ENGINEERING, BANGALORE

BANGALORE 560 035

ABSTRACT

Our project includes the data visualization and prediction of data for a dataset. We have taken our dataset from kaggle and our dataset name is Google Play Store Apps. It is a Web scraped data of 10 thousand Play Store apps for analysing the Android market. In this dataset we have 10841 rows and 13 columns. We have first analyzed and explored the data by plotting various variables. And we have got many applications in play store where it is difficult to predict their ratings. To make it easy and accurate we have predicted the ratings of apps through machine learning algorithms like linear regression and random forest regression. And from the regression results, we opted the best score and concluded the best result of both the app ratings obtained using ML algorithms where we have considered some of the variables from the dataset like No of Downloads, reviews, size and more for predicting.

DATA SET

In the current dataset we have 10841 rows and 13 columns. The columns of the dataset are as follows:

The columns of the dataset are as follows:

a) App: name of the app
b) Category: which category does the app belong to
c) Rating: rating of the app given by the users
d) Reviews: reviews given by the user
e) Size: size of the app in mb or gb
f) Installs: It tells about how many times the app has been developed
g) Type (Free/Paid): it tells that either the app is free or should we buy the app
h) Price (App): if it is paid it shows the price
i) Content Rating (Everyone/Teenager/Adult): It tells the app
j) should be used by everyone or teenager or only for adult
k) Genres (Detailed Category): A app can be divided into different genres according to the categories
l) Last Updated (App): It shows when was the app last updated
m) Current Version (App): Current version of the app available on Play Store
n) Android Version (Support): Min required Android version

Kaggle link for the dataset: https://www.kaggle.com/lava18/google-play-store-apps

MACHINE LEARNING THEORY

Linear regression:

Linear regression is perhaps one of the most well-known and well understood algorithms in statistics and machine learning. Linear regression attempts to model the relationship between two variables by fitting a linear equation to observed data. One variable is considered to be an explanatory variable, and the other is considered to be a dependent variable. A scatterplot can be a helpful tool in determining the strength of the relationship between two variables.

Using linear regression, we are predicting the app ratings using python packages.

Random forest:

Random forests are an ensemble learning method for classification, regression and other tasks that operate by constructing a multitude of decision trees at training time and outputting the class that is the mode of the classes (classification) or mean/average prediction (regression) of the individual trees. The results of random forest determine the importance of all the variable and their influence on the rating. Random forest model is the first model that is applied to the dataset and the results of Random forest classification are computed for a number of variables to find the importance of these variables. However, data characteristics can affect their performance.

Using random forest regression, we are predicting the app ratings using python packages.

IMPLEMENTATION STEPS

1. Firstly, we'll have to download the dataset from Kaggle and then load your downloaded dataset.

2. Before loading your dataset, import the required libraries in your python notebook as follows:

```
import numpy as np # linear algebra
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
data_play = pd.read_csv("googleplaystore.csv")
data_play
```

3. Basic information retrieval is done to anlayze the given data.

4. In the location 10472 there is a data mismatch, inorder to correct it we'll have to edit it as follows using (.loc).

```
dataplay.loc[10472, "Category"] = "LIFESTYLE"
dataplay.loc[10472, "Rating"] = 1.9
dataplay.loc[10472, "Reviews"] = 19
dataplay.loc[10472, "Size"] = "3.0M"
dataplay.loc[10472, "Installs"] = "1000+"
dataplay.loc[10472, "Type"] = "Free"
dataplay.loc[10472, "Price"] = "0"
dataplay.loc[10472, "Content Rating"] = "Everyone"
dataplay.loc[10472, "Genres"] = "Lifestyle"
dataplay.loc[10472, "Last Updated"] = "February 11, 2018"
dataplay.loc[10472,"Current Ver"] = "1.0.19"
dataplay.loc[10472, "Android Ver"] = "4.0 and up"
```

5. There are some outliers in the data, so in order to fill the empty cells of 'Type' and 'Android version', we have used mode for fillna(). And for the remaining columns having empty cells, we have used median for fillna().

6. In price "$" is removed.

   In size, "M", "k" are removed.

   In installs, " + ", " , " are removed.

   Thus, we've converted price, size and installs into integer datatype.

   And also reviews were converted into integer datatype.

   Last updated is converted into datetime format.

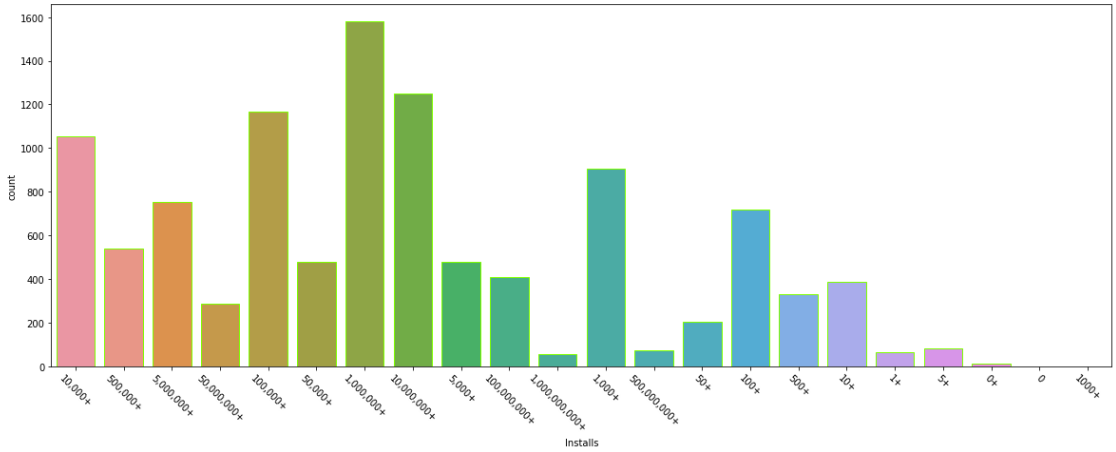
7. We have explored and analyzed our dataset by plotting some of the variables as bar graphs, pie charts and box plots.

   So here is one of the analyzed data:

```
plt.figure(figsize=(20,7))
sns.countplot(dataplay['Installs'], edgecolor = "#7FFF00")
plt.xticks(rotation = -45)
plt.show()
```

```
/usr/local/lib/python3.6/dist-packages/seaborn/_decorators.py:43: FutureWarning:

Pass the following variable as a keyword arg: x. From version 0.12, the only valid positional argument will be `data`, and passing other arguments without an
```



8. In order to predict the rating using ML learnings, the following need to be done:

   From the above analyzed data, we have have splitted the dataset into X and y where X consists of columns other than ‘Reviews’ whereas y consists of the only column ‘Reviews’. Now this X and y are again splitted into training data and test data as follows: X_train, X_test, y_train, y_test.

```
from sklearn.model_selection import train_test_split
X = datamodel.loc[:, datamodel.columns != "Rating"]
y = datamodel['Rating']
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size = 0.3, random_state = 10)
```

9. Using the above information, we predict the rating using linear regression and random forest regression.

10. After predicting the ‘Ratings’ using both the regressions, we also calculate prediction scores inorder to opt the best prediction model for app ratings.

SAMPLE CODE AND OUTPUT

Linear regression code and output:

```
#Let's make the linear model
from sklearn.linear_model import LinearRegression
lr = LinearRegression()
model = lr.fit(X_train, y_train)
print(model)
y_predict = model.predict(X_test)
y_predict
```

```
LinearRegression(copy_X=True, fit_intercept=True, n_jobs=None, normalize=False)
array([4.16444311, 4.1304944 , 4.28767842, ..., 4.18492275, 4.16668494,
       4.1350108 ])
```

```
predict_dataframe = pd.DataFrame(data={"Predicted": y_predict, "Actual": y_test})
predict_dataframe
```

	Predicted	Actual
212	4.164443	4.1
6547	4.130494	4.4
2378	4.287678	4.5
5744	4.073981	4.2
3793	4.164898	4.2
...	...	...
4965	4.203828	4.3
3001	4.243882	3.3
2811	4.184923	4.7
3777	4.166685	3.9
4461	4.135011	4.1

3253 rows × 2 columns

```
model.score(X_test, y_test)
```

```
0.041207418430928455
```

Random forest regression code and output:

```
[ ]  #Lets make Random Forest Regresor
     from sklearn.ensemble import RandomForestRegressor
     rfr = RandomForestRegressor()
     modef_rfr = rfr.fit(X_train, y_train)
     y_predict_rfr = modef_rfr.predict(X_test)
     rfr_df = pd.DataFrame(data={"Predicted": y_predict_rfr, "Actual": y_test})
     rfr_df
```

	Predicted	Actual
212	4.026	4.1
6547	4.592	4.4
2378	4.360	4.5
5744	3.848	4.2
3793	4.477	4.2
...	...	...
4965	4.500	4.3
3001	3.894	3.3
2811	4.393	4.7
3777	4.153	3.9
4461	4.151	4.1

3253 rows × 2 columns

```
[ ]  #Accuracy
     modef_rfr.score(X_test, y_test)
```

```
0.13004103973855763
```

Comparing the scores of Linear regression and Random forest regression:

```
[ ]  #Model Scores and Conclusion
     print("Linear Regression Score: ", model.score(X_test, y_test))
     print("Random Forest Regressor Score: ", modef_rfr.score(X_test, y_test))
```

```
Linear Regression Score:  0.041207418430928455
Random Forest Regressor Score:  0.13004103973855763
```

CONCLUSION

- Thus, we have done the data analysis of many features of the Google Play Store Apps which include installs, reviews, size, prices, ratings, payment, versions, etc.
- App ratings were predicted using Linear Regression model.
- App ratings were predicted using Random Forest Regression model.
- Both the models were finally compared using model scores, opting the best model for prediction of App Ratings in Google Play Store.
- And regarding further approaches, we could do the prediction using Ridge regression and Support vector regression as well.